MODÜL 12

DOSYALARLA ÇALIŞMAK

Şekil 12.1: Bölümle ilgili örnek uygulamalara karekod' dan ulaşabilirsiniz

Bu bölüme kadar Python ile temel programlama kavramları ile tanışıp örnekler üzerinden uygulamalar yaptınız. Koşul yapısı, döngüler listeler fonksiyonlar gibi kavramlar ile tanışıp program yazarken kullandınız. Ayrıca kullanıcıdan veri alıp işlediniz ve bir sonuç / çıktı ürettiniz. Peki elde ettiğiniz sonuç/çıktıları kalıcı olarak depolayabildiniz mi veya depolanmış veriyi programınıza dâhil edebildiniz mi?

Bu bölümde elde ettiğiniz çıktıları dosyaya nasıl kaydedebileceğinizi, dosyadan nasıl veri okuyacağınızı ve iki işlemi birlikte nasıl yapabileceğinizi öğreneceksiniz. Ayrıca var olan dosyalara erişip üzerinde güncelleme yapmayı öğreneceksiniz. **Bu bölümdeki uygulamalar idle ile yazılmıştır. Oluşturulan dosyalar ve kodlar aynı klasöre kaydedilmiştir.**

Dosyalar verilerin kalıcı olarak depolandığı ve ihtiyaç duyulduğunda okunabildiği temel yapılardandır. Verilerin depolanması ve üzerinde işlemler yapılması için kullanılan gelişmiş yapılar veritabanlarıdır. Veritabanı konusu modül on dörtte işlendiğinden bu bölümde sadece dosyalar ile ilgili kısmına değinilmiştir.

Dosyalarla çalışma mantığı ve süreci Şekil 12.2'de gösterildiği gibi üç aşamadan oluşmaktadır. Öncelikle dosya amaca uygun kipte açılır. Ardından veri okuma-yazma gibi işlemler yapılır. Son olarak dosya kapatılır.

Şekil 12.2: *Doyalar ile çalışma süreci*

12.1. Dosya İşlemleri

Python'da dosya işlemleri Python'un içinde bulunan hazır nesneler ile yürütülür. Bunun için metin sarma (text wrapper) nesnesi oluşturulur.

dosya=open(dosya adı, dosya açma türü)

Python'da bir dosyayı açmak için **open( )** fonksiyonu kullanılır. open fonksiyonu içine dosya adı ve erişim türü girdilerini alır. Bunu şöyle düşünebilirsiniz:

Hangi dosyayı açayım? Hangi amaç için açayım?

Hangi dosyayı açayım sorusunun cevabı adresini belirttiğiniz dosyadır. Bir dosya hangi amaçlar için açılır derseniz bunlar:

veri okuma, veri yazma ve ikisinin (okuma ve yazma) birlikte olduğu durumlardır.

dosya ile işimiz bittiği zaman dosyayı **close()** fonksiyonu ile kapatırız.

Ek bilgi olarak karakter kodlaması içinde 3. parametre olarak encoding kullanılır. Türkçe karakterlerde sorun yaşamamak için **encoding="utf-8"** ifadesi eklenir.

Dosya işlemlerini yaparken 2 farklı yöntem kullanabiliriz. Tablo 1'de gösterildiği gibi dosyayı kendimizin açıp kendimizin kapattığı yöntem ya da **with open** parametresi kullanılarak dosyanın sürecin bitiminde otomatik kapandığı yöntem kullanılabilir. İki yöntemi de kullanabilirsiniz.

Tablo 1: Dosya işlemleri süreci yöntemleri

dosya=open (dosya adı, açma kipi) işlemler dosya.close ()	with open(dosya adı, açma kipi) as dosya: **işlemler** # girinti kullanmaya dikkat

Dosya işlemlerine başlamadan önce ilerleyen örneklerde kullanılmak üzere deneme.txt, veri.txt gibi dosyaları Python kodlarının olduğu ve .py uzantılı olan kod dosyası ile aynı klasörde oluşturunuz. Böylece dosyanın sadece adını yazarak bilgisayarda bulunduğu yeri yazmaya ihtiyaç duymadan açabiliriz. Peki ya, dosyayı hangi amaç için açacağız ve amacımıza göre açma yöntemi hangisidir? derseniz cevabı dosya kipleri olacaktır.

12.2. Dosya Kipleri

Dosyalara erişme amaçlarımız veri okuma, yazma ve iki amacın birlikte olduğu durumlardır. Bu amaçları karşılamak üzere Python'da dört farklı dosya açma kipi bulunmaktadır.

1. Dosyadan sadece veri okumak istiyor isek **r** (read) kipinde açarız. Bu kipte yazma işlemi yapmaya çalışırsak hata ile karşılaşırız.
2. Dosyaya sadece veri kaydetmek/yazmak istiyor isek **w** (write) kipinde açarız. Bu kipte okuma işlemi yapmaya çalışırsak hata ile karşılaşırız.
3. Dosyadan hem veri okumak hem de dosyaya veri yazmak istiyor isek **r+** kipinde açarız. Bu kipte dikkat etmemiz gereken nokta dosya oluşturulmamış ise hata ile karşılaşırız.
4. Dosyadan hem veri okumak hem de dosyanın sonuna veri yazmak istiyor isek **a** (append) kipinde açarız. a kipinde dosyayı açtığımızda dosya yok ise oluşturur r+ olduğu gibi hata vermez.

Tablo 2'de dosya okuma kipleri ve özellikleri hakkında karşılaştırmalı ve detaylı bilgiler verilmiştir.

Tablo 2: Dosya okuma kipleri ve özellikleri

Dosya açma kipi	Dosya açma kodu	Dosya mevcutsa ve yoksa ne olur
yazma kipinde dosya açma w (write)	dosya=open("deneme.txt","w") işlemler dosya.close()	yoksa oluşturulur. var ise dosya oluşturulur ve eski dosya silinir. Eski dosyada bilgi var ise gider.
ekleme kipinde dosya açma a (append) (ekleme)	dosya=open("deneme.txt","a") işlemler dosya.close()	yoksa oluşturulur. var ise dosya oluşturulmaz. dosyanın sonuna ekleme yapılır. eski bilgiler silinmez. Sadece okuma işlemi yapamazsınız.
okuma kipinde dosya açma r (read)	dosya=open("deneme.txt","r") işlemler dosya.close()	yoksa hata üretilir. var ise okunur
Okuma ve yazma (birlikte) Dosya açma r+	dosya=open("deneme.txt","r+") işlemler dosya.close()	yoksa hata üretilir. var ise dosya oluşturulmaz dosyanın istenen bölümlerine eklemeler yapılır. eski bilgiler silinmez

12.3. Dosya Okuma Yazma İşlemleri

Bu bölümde dosya okuma ve yazma örnekleri bulunmaktadır. Öncelikle deneme.txt isimli bir dosya oluşturarak içine "bu basit bir dosyadır" cümlesini yazmalısınız. Örnekleri uyguladıktan sonra deneme.txt dosyasında nelerin değiştiğini görmek için dosyayı açmalısınız. Dosyayı açıp inceledikten sonra kapatmayı unutmayınız.

NOT

"\n" ifadesi print( ) örneklerinden hatırlayacağınız gibi alt satıra geçmek için kullanılmaktadır.

İlk olarak oluşturduğunuz deneme.txt dosyası okuma kipinde açılacak ve içindeki verileri ekranda görüntülenecektir.

Örnek 1

Dosyayı okuma kipinde açma:

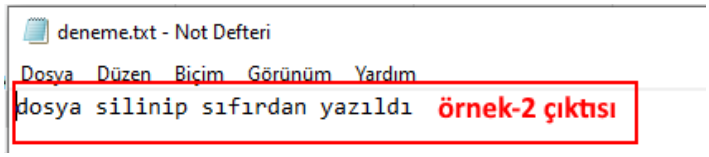
```
dosya = open ("deneme.txt","r")
belge=dosya.read()
print (belge)
dosya.close()
bu basit bir dosyadır
```

İkinci aşamada deneme.txt dosyası yazma kipinde açılıp içine veri kaydedilecektir. Yazma işlemini yaptıktan sonra Resim 12.3'te görüldüğü gibi dosyada önceden bulunan verilerin silindiğini fark ettiniz mi?

Örnek 2

Dosyayı yazma kipinde açma:

```
with open ("deneme.txt","w") as dosya:
dosya.write ("dosya silinip sıfırdan yazıldı ")
```



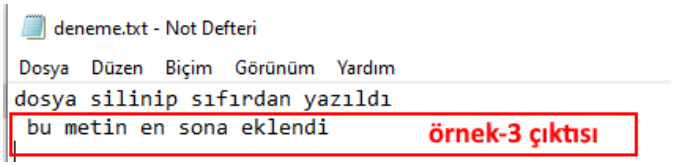
Resim 12.3: *Örnek 2 kod çıktısı*

Şimdi de dosyayı okuma ve yazma kipinde açıp neler olduğunu inceleyiniz.

Örnek 3

Dosyayı okuma ve yazma kipinde açma:

```
dosya = open("deneme.txt","r+")
belge=dosya.read()
print(belge)
dosya.write("\n bu metin en sona eklendi")
dosya.close()
dosya silinip sıfırdan yazıldı
```



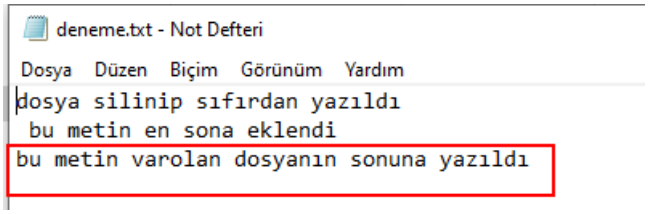
Resim 12.4: *Örnek 3 kod çıktısı*

Son olarak dosyayı genişleme kipinde açıp neler olduğunu inceleyiniz.

Örnek 4

Dosyayı genişletme kipinde açma:

```
dosya = open ("deneme.txt","a")
dosya.write ("bu metin dosyanın sonuna yazıldı")
dosya.close()
```



Resim 12.5: *Örnek 4 kod çıktısı*

Python'da dosyalara yazılan veriler string (metin) türünde kaydedilmektedir. Dosya erişim örneklerini öğrendikten sonra programlamada temel algoritmalar olarak kabul edilen arama ve sıralama algoritmalarını kullanarak dosya işlemleri yapılabilir.

12.4. Arama ve Sıralama Algoritmaları ile Dosyadan Okuma ve Dosyaya Yazma

İlk örnek şifreleme (kriptoloji) ve şifre çözme biliminde önemli bir yeri olan asal sayı örneği olacaktır. Çok büyük sayıların asal olup olmadığının kontrolü için bilgisayara hesaplatmak yerine kayıtlı olan bir listeden bakmak daha kolay bir yöntemdir. Bu nedenle bunun bilgisayarı yormayacak uzunlukta bir örneği yapılacaktır. Birinci aşamada birden bine kadar olan asal sayılar hesaplanıp asal.txt isimli dosyaya kaydedilecektir. Diğer bir değişle dosya yazma kipinde açılacaktır. Bu durumda yapılacak olan işlem asal sayıların hesaplanıp dosyaya kaydedilmesi olacaktır. Asal sayı örneği döngüler konusunda anlatıldığından ek bir açıklamaya gerek duyulmamıştır. Dosya açma kipi: yazma kipi olacak. Asal sayılar bulunduktan sonra string türüne dönüştürülmüştür. Sayılar arasına bir boşluk bırakılmıştır. Böylece Örnek 6'da dosyadan asal sayılar okunduktan sonra split metodu ile sayıları liste veri tipinde tutulması sağlanmış olacaktır. Aynı mantık virgül gibi başka bir karakter kullanılarak yapılabilir.

Örnek 5

1'den 1000'e kadar olan asal sayıları bulup dosyaya kaydeden program.

```
asal_sayı=[2]
for sayı in range (3,1001):
    for bolen_sayı in range (2,sayı):
        sayı_asalmı=False
        if sayı % bolen_sayı==0:
            sayı_asalmı=True
            break
    if sayı_asalmı==False:
            asal_sayı.append(sayı)
```

```
veri=" "
for i in asal_sayı:
    veri+=str(i) # veri=veri+str(i)
    veri+=" "
dosya=open ("asalsayı.txt","w")
dosya.write(veri)
dosya.close()
asalsayı.txt nin içindeki veriler çok yer kaplayacağından buraya eklenmemiştir.
Dosyayı açıp sonucu inceleyebilirsiniz.
```

Asal sayılar bulunup dosyaya kayıt edildikten sonra kullanıcının girdiği sayının asal olup olmadığı dosyadan kontrol edilmektedir. Bunun için öncelikle dosya açılıp veriler okunmaktadır. Ardından okunan veriler split metodu kullanarak bir listeye kaydedilmektedir. Böylece kullanıcının girdiği sayının listede olup olmadığına bakılarak sayının asal olup olmadığını kontrol edilmiş olacaktır.

Örnek 6

Girilen sayının asal olup olmadığını dosyadan karşılaştıran program

```
with open("asalsayı.txt","r") as dosya :
    veri=dosya.read()
    asal_sayılar=veri.split(" ")
kontrol_sayısı=input("asal olup olmadığını kontrol etmek istediğiniz sayıyı
giriniz")
if kontrol_sayısı in asal_sayılar :
    print("asal sayı")
else:
    print("asal sayı değil")
asal olup olmadıgını kontrol etmek istediğiniz sayıyı giriniz22
asal sayı değil
```

Aklınıza bu iki örneği tek bir programda birleştirebilir miyiz? sorusu gelebilir. Örnek 7'de dosyaya kaydetme ve dosyada arama işlemlerinin birlikte yapılabileceği bir rezervasyon uygulaması yapılacaktır. Uygulamada rezervasyon yapma ve rezervasyon sorgulama işlemleri bulunmaktadır. Bu işlemler birer fonksiyon olarak tanımlanıp örnek programda kullanılmaktadır.

Örnek 7

Rezervasyon yapan ve rezervasyon kaydını kontrol eden program

```
def rezerevasyon_yap():
    rezervasyon_bilgi=input("rezervasyon bilgilerinizi giriniz")
    veri=rezervasyon_bilgi+","
    dosya=open("rezervasyon.txt","a")
    dosya.write(veri)
    dosya.close()

def rezerevasyon_kontrol() :
    rezervasyon_bilgi=input("rezervasyon bilgilerinizi giriniz")
    with open("rezervasyon.txt","r") as dosya:
        veri=dosya.read()
        rezervasyonlar=veri.split(",")
        if rezervasyon_bilgi in rezervasyonlar:
            print("rezervasyonunuz bulunmaktadır.")
        else:
            print("rezervasyon kaydınız yoktur.")

while True:
    islem=input("rezervasyon yapmak için  1 : kontrol için 2 programı kapatmak için
3 e basınız")
    if islem=="1" :
        rezerevasyon_yap()
    elif islem =="2":
        rezerevasyon_kontrol()
    else:
        break
```

Yukarıdaki örneği yaptıktan sonra aklınıza şu soru gelebilir peki yapılan dosyaya kaydedilen rezervasyon bilgileri üzerinde değişiklik (düzeltme, silme gibi) işlemleri yapılabilir mi? Bunun örneği ilerleyen örneklerde yapılacaktır.

12.5. Dosyaların Özel Metotları

read( ) fonksiyonunu dosyanın tamamını okumak için kullandık. Aynı metot dosyanın belirli bir kısmını okumak için de kullanılabilir. Örneğin, **read(10)** ile 10 byte veri okunur. Ayrıca dosyadan veri okurken for döngüsünün de kullanıldığını belirtmekte yarar var. Bir önceki bölümde anlatılan fonksiyonlarının dışında **seek(), writeline(), tell()** ve **writelines()** metotları bulunmaktadır.

seek() fonksiyonu imlecin dosyada istenen noktaya alınması için kullanılır. **tell()** fonksiyonu da imlecin güncel olarak nerede bulunduğunu öğrenmek için kullanılır. Microsoft word gibi bir kelime işlemci programından aşina olduğunuz imleç yazma veya okuma işlemlerinde aktif olan konumu ifade eder. Örnek 8'de tell(), seek(), read(girilen bayt) ve for döngüsü ile dosya okuma örneği verilmiştir.

Örnek 8

tell(), seek(), read(girilen bayt) ve for döngüsü ile dosya okuma örneği:

```
dosya=open("deneme.txt","r")
# dosyamızı for döngüsü ile okuyoruz
for veri in dosya:
    print(veri)
#imlecin nerede olduğunu ekrana yazdırıyoruz
print(dosya.tell())
# imleci 10. bayta taşıyoruz
dosya.seek(10)
#imlecin bulunduğu yerden 20 bayt veri okuyoruz
print(dosya.read(20))
#imlecin nerede olduğunu görüntülüyoruz
print(dosya.tell())
dosya.close()
bu basit bir dosyadırbu metin en sona eklendibu metin varolan dosyanın sonuna
yazıldı
85
ir dosyadırbu metin
30
```

Dosyalarda satır satır işlem yapabilmek için verileri de farklı satırlarda olacak şekilde kaydetmeniz gerekir. Bunu yapmanın en yaygın yolu verinin sonuna "\n" eklemektir.

writelines() fonksiyonu string içerikli listeleri dosyaya yazar. **readline()** fonksiyonu bir satır veri okur.

readlines() fonksiyonu dosyanın tamamını okuyup liste veri tipinde tutar. Böylece liste metodları kullanılabilir. Örnek 9-12'de bu fonksiyonların kullanımına dair örnek uygulamalar yapılmıştır.

Örnek 9

tell(), seek(), writelines() uygulaması

```
dosya=open("deneme.txt","r+")
dosya.seek(20)
#dosyada 20. bayta gittik
dosya.write("20. bayttan itibaren yazdık")
#20. bayttan sonraki verilerin üzerine yazdık
print(dosya.tell())
# imlecin 47. bayta geldiğini öğreneceğiz
#burda dosyadan okuma yaparsak 47. bayttan sonrası okunacak
print(dosya.read())
# listemizdeki verileri en sona yazıyoruz
liste=["1","2","3","4"]
#listedeki veriler string olmazsa hata alırız.
dosya.writelines (liste)
47
metin varolan dosyanın sonuna yazıldı
```

Örnek 10 ve 11 için İstiklal Marşı'nın 10 kıtasını kodlarla aynı klasörde olacak şekilde istiklal.txt isimli bir dosyaya kayıt ediniz.

Örnek 10

Dosya metotları uygulaması

```
dosya=open("istiklal.txt","r",encoding="utf-8")
# readlines ile içeriği liste olarak alacam
liste=dosya.readlines()
# listenin uzunluğunu hesaplayarak kaç satır olduğunu ekrana yazıyorum
print(len(liste))
#listenin 6. satırını ekrana yazıyorum
#listenin ilk satırının liste[0] olduğunu hatırlayın
print(liste[5])
#dosyada başa dönüyorum
dosya.seek(0)
# baştan sona satır satır okuyorum
for satır in range(len(liste)):
    print(dosya.readline())
dosya.close()
41
Kahraman ırkıma bir gül! Ne bu şiddet, bu celal?
İstiklal marşının 10 kıtasının tamamı ekrana yazılıyor (çok yer kaplamaması adına
yazılmamıştır)
```

Örnek 11'de dosyanın içinde belirli konumlara erişip veri ekleme örneği bulunmaktadır.

Örnek 11

Dosyanın içinde belirli yerlere veri ekleme

```
dosya=open("istiklal.txt","r+",encoding="utf-8")
# readlines ile içeriği liste olarak alacam
liste=dosya.readlines()
print(len(liste))
# her 4 kıtadan sonra araya bir satır çizgi ekliyoruz
# listeye 4 satırda bir ek satır eklediğimden beşer beşer ilerleyeceğim
eklenecek_satır=30*"-"+"\n"
for ekle in range(4,52,5):
    liste.insert(ekle,eklenecek_satır)
# dosyada başa gelecem
dosya.seek(0)
# çizgi eklenmiş listeyi yazdıracağım
dosya.writelines(liste)
dosya.seek(0)
yeni_liste=dosya.readlines()
print(yeni_liste)
dosya.close()
sonuç çok yer kaplayacağından burada gösterilmemiştir. Lütfen istiklal.txt
dosyasının inceleyiniz
```

Örnek 12'de özel fonksiyon kullanımını içeren rezervasyon uygulaması bulunmaktadır.

Örnek 12

Rezervasyon kapasite kontrolü uygulaması

```
dosya=open("rezervasyon.txt","r+",encoding="utf-8")
rezervasyonlar=dosya.readlines()
rezervasyon_kapasitesi=50
sıra_no=1
for rezervasyon_sahibi in rezervasyonlar:
    print(sıra_no,"nolu rezervasyon sahibi",rezervasyon_sahibi)
    sıra_no+=1
print("toplam",sıra_no,"adet rezervasyon var")
if rezervasyon_kapasitesi-sıra_no >0 :
    print(rezervasyon_kapasitesi-sıra_no,"adet daha  rezervasyon yapabiliriz")
else:
    print("rezervasyon kapasitemiz dolmuştur.")
dosya.close ()
```

Örnek 13'te özel fonksiyonlar ile rezervasyon yapma uygulaması örneği bulunmaktadır.

Örnek 13

Özel fonksiyonlar ile rezervasyon yapma uygulaması

```
def rezervasyon_yap():
    rezervasyon_kapasitesi=50
    dosya=open("rezervasyon.txt","r+", encoding="utf-8")
    rezervasyonlar=dosya.readlines()
    mecut_rezervasyon=len(rezervasyonlar)
    if rezervasyon_kapasitesi-mecut_rezervasyon >0 :
        print(rezervasyon_kapasitesi-mecut_rezervasyon,"adet daha  rezervasyon
yapabiliriz")
        yeni_rezervasyon=input("rezervasyon bilgilerini araya virgül girerek
giriniz")
        dosya.write("\n"+yeni_rezervasyon )
        print("rezervasyon numaranız =",mecut_rezervasyon+1,end="")
        print("rezervasyon bilgileriniz : ",yeni_rezervasyon)
        print("razervasyonunuz başarıya tamamlanmıştır.")
    else:
        print("rezervasyon kapasitemiz dolmuştur.")
    dosya.close()

print("rezervasyon ekranına hoş geldiniz")
rezervasyon_yap()
```

Örnek 14'te rezervasyon güncellemesi içeren dosya uygulaması bulunmaktadır.

Örnek 14

Rezervasyon güncelleme uygulaması

```
def rezervasyon_güncelle(rezervasyon_no):
    dosya=open("rezervasyon.txt","r+",encoding="utf-8")
    rezervasyonlar=dosya.readlines()
    if rezervasyon_no <len(rezervasyonlar):
        print("rezervason bilgileriniz: ",rezervasyonlar[rezervasyon_no])
        güncelleme=input(" lütfen rezervasyon güncelleme bilgilerini giriniz")
        rezervasyonlar[rezervasyon_no]=güncelleme+"\n"
        print(" güncel rezervason bilgileriniz: ",rezervasyonlar[rezervasyon_no])
        dosya.seek(0)
        dosya.writelines(rezervasyonlar)
        dosya.close()
        print("rezervasonunuz başarıyla güncellenmişitir.")

    else:
        print("hatalı bir rezervason numarası girdiniz")

print("rezervasyon güncelleme ekranına hoş geldiniz")
rezervasyon_no=input("lütfen rezervasyon numaranızı giriniz")
rezervasyon_güncelle (int(rezervasyon_no))
```

Örnek 15'te yapılan rezervasyonun nasıl silineceğini gösteren bir dosya uygulaması bulunmaktadır.

Örnek 15

Rezervasyon silme uygulaması

```
def rezervasyon_sil(rezervasyon_no):
    dosya=open("rezervasyon.txt","r+",encoding="utf-8")
    rezervasyonlar=dosya.readlines()
    if rezervasyon_no <len(rezervasyonlar):
        print("rezervason bilgileriniz: ",rezervasyonlar[rezervasyon_no])
        emin_misiniz=input("kaydı silmek istediğinziden emin misiniz e/h")
        if emin_misiniz=="e" or emin_misiniz=="E":
            rezervasyonlar.pop(rezervasyon_no)
            dosya.seek(0)
            dosya.writelines(rezervasyonlar)
            dosya.close()
            print("rezervasonunuz başarıyla güncellenmişitir.")

        else:
            print("rezervasyon silme işleminiz iptal edilmişitir")

    else:
        print("hatalı bir rezervason numarası girdiniz")

print("rezervasyon silme ekranına hoşgeldiniz")
rezervasyon_no=input("lütfen rezervasyon numaranızı giriniz")
rezervasyon_sil(int(rezervasyon_no))
```

12.6. Bölüm Sonu Örnekleri

1. Elli adet rastgele sayı üretip sayı.txt dosyasına kaydeden program uygulaması
2. Rastgele üretilen sayıları alıp büyükten küçüğe doğru sıralayıp sıralı.txt dosyasına kaydeden program uygulaması

Cevaplar

1.
```
import random
sayılar=[]
for i in range (50):
    sayı=str(random.randint(0,1000))+"\n"
    sayılar.append(sayı)
dosya=open("sayı.txt","w")
dosya.writelines(sayılar)
dosya.close()
```

2.
```
dosya=open("sayı.txt","r")
sayılar=dosya.readlines()
dosya.close()
# sayıları string olarak aldık sayıya dönüştürüp sıralayıp tekrardan stringe
dönüştüreceğiz
for i in range(len(sayılar)):
   sayılar[i]=int (sayılar[i])
   sayılar.sort()
for i in range(len(sayılar)):
   sayılar[i]=str(sayılar[i])+"\n"
dosya=open("sıralı.txt","w")
dosya.writelines(sayılar)
dosya.close()
```